Trabajo Práctico Obligatorio: **Uno Mas**

Condiciones generales

- Armar grupos de hasta 5 (cinco) personas.
- Se debe entregar un documento PDF con todas las consignas solicitadas. El mismo deberá contener la carátula con el nombre, apellido y LU de los integrantes del equipo.
- Fechas de entregas:
    - Primera entrega: ver cronograma
    - Entrega final: ver cronograma
- En los días de entrega se debe exponer la solución propuesta.
    - Si bien el trabajo práctico es grupal, **la evaluación es individual y lleva nota.**

Introducción

Se solicita el desarrollo de un sistema para la gestión de encuentros deportivos en el que los usuarios puedan encontrar jugadores para completar un equipo de fútbol, básquet, vóley o cualquier otro deporte disponible.

El sistema deberá seguir el patrón arquitectónico MVC y aplicar al **menos cuatro** de los siguientes patrones de diseño: **Strategy**, **Adapter**, **State**, **Observer**, **Composite**, **Decorator**, **Factory** y **Facade**.

Requerimientos Funcionales

1. **Registro de usuarios:** Los usuarios deberán registrarse en la aplicación móvil mediante un nombre de usuario, correo electrónico y contraseña. Opcionalmente, podrán ingresar su deporte favorito y nivel de juego.
2. **Búsqueda de partidos**: Los usuarios podrán buscar encuentros deportivos en su zona donde falten jugadores.
3. **Creación de un partido**: Un usuario podrá crear un partido en el sistema y definir los siguientes atributos:
    a. Tipo de deporte
    b. Cantidad de jugadores requeridos
    c. Duración del encuentro
    d. Ubicación y horario
    e. El encuentro recién creado pasará a estar en el estado inicial: "**Necesitamos jugadores**"
4. Estado de los partidos:
    a. Cuando un partido alcance el número requerido de jugadores, pasará automáticamente al estado "**Partido armado**".
    b. **Confirmado**: Todos los jugadores aceptaron la participación y el partido está listo para jugarse.
    c. **En juego**: El partido ha comenzado. Se transiciona automáticamente cuando llega la fecha y hora del encuentro.
    d. **Finalizado**: El partido ha concluido y se pueden registrar estadísticas o comentarios.
    e. **Cancelado**: El organizador cancela el partido antes de su inicio.
5. Estrategia de emparejamiento de jugadores:
    a. Se implementará una estrategia de emparejamiento basada en el nivel de juego de los usuarios.
    b. Los jugadores podrán tener distintos niveles ("Principiante", "Intermedio" o "Avanzado").
    c. Los partidos pueden configurarse para permitir jugadores de cualquier nivel o establecer un mínimo/máximo de nivel requerido.

   d. Se podrá definir diferentes algoritmos de emparejamiento (por cercanía, por nivel de habilidad o por historial de partidos previos).
6. Notificaciones:
   a. Se enviarán notificaciones push a través de Firebase y correos electrónicos (aún no se conoce la librería a utilizar, inicialmente puede usar JavaMail) a los jugadores cuando:
      i. Se cree un partido nuevo para su deporte favorito.
      ii. Se unan suficientes jugadores y el partido pase a estado "**Partido armado**".
      iii. Se confirme el partido.
      iv. El partido cambie a estado "En juego", "Finalizado" o “Cancelado”.

Requerimientos No Funcionales

- La aplicación debe seguir el patrón MVC.
- Se debe usar al menos cuatro patrones de diseño de los listados anteriormente.
- Se debe presentar un diagrama de clases UML con los patrones identificados.
- Se debe desarrollar la implementación en código del sistema.

Entregables

Se deberá presentar:

- Diagrama de clases UML con la aplicación de los patrones de diseño.
- Explicación breve de cómo se implementaron los patrones en el sistema.
- Código fuente con la implementación en un archivo comprimido. Se recomienda implementar el código en Java, pudiendo optar por otro lenguaje orientado a objeto que sea de su preferencia.

Evaluación

La evaluación se basará en:

- Correcta aplicación del modelo MVC.
- Implementación de los patrones de diseño.
- Claridad y organización del código.
- Funcionalidad del sistema.